

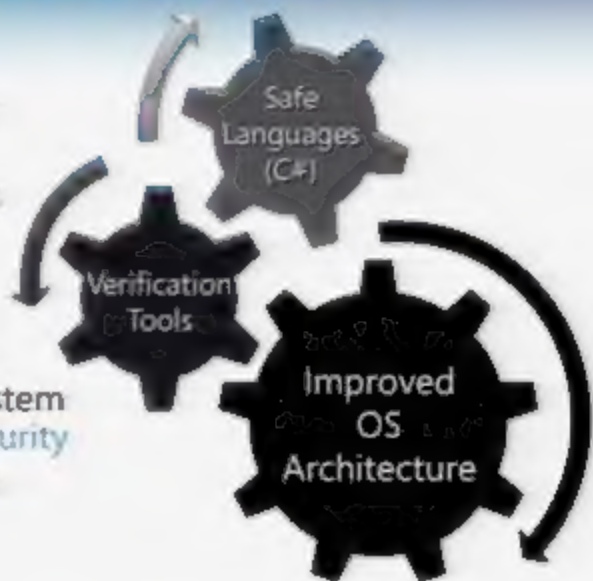
Singularity In A Nutshell

Galen C. Hunt, Ph.D.
Principal Researcher, OS Group
Microsoft Corporation

Singularity

Building dependable software systems

- Research OS
 - not a product
 - not a replacement for Windows
 - test bed for new ideas
- Goal is to
 - overcome faults and limits of current OS's
 - drastically improve system dependability and security
 - achieve good enough performance



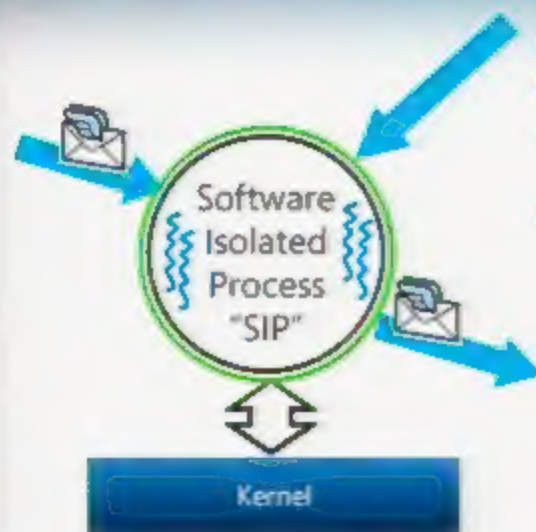
Architecture

Founded on static verification

- Singularity contributes three key ideas:
 - Software-Isolated Processes (SIPs)
 - Contract-Based Channels (CBCs)
 - Manifest-Based Programs (MBPs)

Software-Isolated Processes

Verified process protection



- Set of code and data pages.
- Data pages contain typed objs
- Code pages contain safe code
 - (type safe and memory safe)
- => No need for HW protection

Contract-Based Channels

Verified inter-process communication

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TopConnectionContract {  
    ...  
    state Connected : one {  
        Read? -> Reading;  
        Write? -> Writing;  
        ...  
    }  
  
    state Reading : one {  
        Data! -> Connected;  
        NoMoreData! -> SendOnly;  
        RemoteClose! -> Zombie;  
    }  
    ...  
}
```

Contract-Based Channels

Verified inter-process communication

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TopConnectionContract {  
    ...  
    state Connected : one {  
        Read? -> Reading;  
        Write? -> Writing;  
        ...  
    }  
  
    state Reading : one {  
        Data! -> Connected;  
        NoMoreData! -> SendOnly;  
        RemoteClose! -> Zombie;  
    }  
    ...  
}
```

```
...  
conn.SendRead();  
switch receive {  
    case conn.Data(readData) :  
        dataBuffer.AddToTail(readData);  
        return true;  
  
    case conn.RemoteClose() :  
        return false;  
}  
...  
}
```

Contract-Based Channels

Verified inter-process communication

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TopConnectionContract {  
    ...  
    state Connected : one {  
        Read? -> Reading;  
        Write? -> Writing;  
    }  
    ...  
    state Reading : one {  
        Data! -> Connected;  
        NoMoreData! -> SendOnly;  
        RemoteClose! -> Zombie;  
    }  
    ...  
}
```

```
...  
conn.FeedRead();  
switch receive {  
    case conn.Data(readData) :  
        databuffer.AddOffset(readData);  
        return true;  
    case conn.RemoteClose() :  
        return false;  
    ...  
}
```

Missing Case
case conn.NoMoreData() :

Contract-Based Channels

Verified inter-process communication

- Contracts define messages and exchange patterns for inter-process communication.

```
contract TopConnectionContract {  
    ...  
    state Connected : one {  
        Read? -> Reading;  
        Write? -> Writing;  
    }  
    ...  
    state Reading : one {  
        Data! -> Connected;  
        NoMoreData! -> SendOnly;  
        RemoteClose! -> Zombie;  
    }  
    ...  
}
```

```
...  
conn.SendRead();  
switch receive {  
    case conn.Data(readData) :  
        dataBuffer.AddToTail(readData);  
        return true;  
    case conn.RemoteClose() :  
        return false;  
}
```

Missing Case

case conn.NoMoreData() :

Contract conformance statically
detects subtle errors.

Manifest-Based Programs

Verified system composition

- Programs declare requirements and services through manifests

```
[DriverCategory]
[Signature("/pci/03/00/5333/8811")]
class S3Trio64Config : DriverCategoryDeclaration
{
    [IoMemoryRange(0, Length = 0x400000)]
    IoMemoryRange framebuffer;

    [IoFixedPortRange(Base = 0x3c0, Length = 0x20)]
    IoPortRange control;

    [ExtensionEndpoint(typeof(ExtensionContract.Exp))]
    TRef<ExtensionContract.Exp;Start> pnp;

    [ServiceEndpoint(typeof(VideoDeviceContract.Exp))]
    TRef<ServiceProviderContract.Exp;Start> video;
}
```

driver for PCI Device

HW must provide
4MB frame buffer

System must
provide channel to
I/O manager

Driver provides
video service

Singularity Version 1.0



- **Micro-kernel like design**
 - Services and Drivers in SIPs
- **Safe Kernel and System**
 - 95% written in C#
 - 17% of files contain unsafe C#
 - 5% of files contain x86 or C++
- **"Complete" Server OS**
 - Runs SPECweb99, etc.
 - TCP/IP stack, file system, etc.
 - Wide classes of device drivers

Singularity Is A Research Vehicle

- We wrote Singularity with the following goal:
 - Imagine you have a new OS research idea, you can produce a strong publishable result in 1 or 2 man-years
- Architecture is clean
 - Simple, orthogonal abstractions
 - Simple implementation

Ongoing Research In MSR

Singularity version 2.0

- A few areas of active research:
 - Compile-Time Reflection
 - Hardware Protection Domains
 - Heterogeneous Many-core Processing
 - Type Assembly Language
 - Post-PC Computing (cPhone)

announcing

Singularity Version 1.0
Research Development Kit

Summary

- Singularity is basis for more dependable systems
 - pervasive use of safe programming languages
 - lightweight, closed, customizable run-time environment
 - verifiable specification of system behavior
- Working research prototype
 - driving research in large number of areas
- For more information:
 - <http://research.microsoft.com/os/singularity>
 - Start with the *Operating Systems Review* paper.
 - RFP: mark.lewin@microsoft.com

Microsoft

Your potential. Our passion.™